• Distributed Scheduling
  – Resource management component of a system which moves jobs around the processors to balance load and maximize overall performance.
  – Typically makes sense in LAN level distributed systems due to latency concerns.
  – Needed because of uneven distribution of tasks on individual processors
    • Can be due to several reasons.
    • Can even make sense for homogeneous systems with (on average) even loads.
• How does one characterize
  – Performance: average response time
  – Load:
    • It has been shown that queue lengths for resources (e.g. CPUs) can be a good indicator.
    • How does one handle the delay of transfer when systems are unevenly loaded and we seek to rectify that?
      – Timeouts, holddowns
    • Queue length not very appropriate for (nor correlated with) CPU utilization for some tasks (e.g. interactive).
• Load balancing approaches may be
  – Static: Decisions are “hard wired” a-priori into the system based on designers understanding.
  – Dynamic: Maintain state information for the system and make decisions based on them. Better than static, but have more overhead.
  – Adaptive: A subtype of dynamic, they can change the parameters they analyze based on system load.

• Load balancing vs. Load sharing
  – Balancing typically involves more transfers. However, sharing algorithms that transfer in anticipation can also cause more transfers.
• Transfers may be preemptive or non-preemptive
  – Preemptive transfers involve transferring execution state as well as the task. Non-preemptive transfers are essentially “placements”

• Load Distribution System Components
  – Transfer policy: Which node should send, who should receive (threshold based approaches are common)
  – Selection policy: Which task should be moved (new tasks, location independent tasks, long running tasks …)
  – Location Policy: Finding a receiver for a task. Typical approaches are polling or broadcast.
  – Information Policy
    • Demand driven, Periodic, or State Change driven
• Stability in a load sharing system
  – Queuing Theoretic: When total work arrival (tasks + load sharing overhead) is greater than rate at which CPU can work. Alternatively, look at the effectiveness of the algorithm.
  – Algorithmic : Does the algorithm lead to thrashing ?